Fusion CI Studios
…innovative, leading-edge dynamic fx

Whole Water

Fluid Simulation Case Study

Final 30-second spot
Final 15-second spot

That Scary/Excited Feeling….

"From the moment we first laid eyes on the storyboard, we knew this was going to be one of the most challenging projects we had ever undertaken at DMG. If we had grasped at the time, just how challenging - we might well have thought twice about saying yes with such unseemly haste. From the outset, we realised we were going to need some specialist help..." Linds Redding, Department of Motion Graphics

The Department of Motion Graphics (DMG) in Auckland, New Zealand approached Fusion CI Studios to collaborate on a commercial project requiring water to do some, well… *unusual* things. Creative agency, Sugar, had come up with an exciting idea for their client Fonterra's new drinking water product "Whole", in which some wildly 'intelligent' water communicates the spot's message.

Photo-real water would leap into the air to form shape after shape: first a brain, then a bridge, then bananas. After the bananas form, all but one splash to the floor, the last one then peels itself and half of the peeled banana splashes to the floor. The creative ideal was for the water to morph between shapes with a vigorous life of its own. When it formed the shapes, they needed to be detailed and sharply defined, while still looking like water, not like geometry with a water shader applied.

The spot's concept was both incredibly exciting and remarkably challenging! Fluid morphing is of course a very cool effect and, by pure coincidence, Fusion had just finished some initial R&D on morphing when DMG contacted us. Our R&D at that point proved we could morph fluid into any geometry, but that was all. The challenge now was to make it morph from shape to shape, to clearly define the complex forms, and to do so with a lot of creative control. Based on our initial work, we were optimistic about this. More of an issue, however, was speed. Our early tests were quite slow with a lot of calculations involved. We'd have to speed up the sim considerably for Whole, because the main shot of the project was 650 frames of continuous water action – and it had to be a *single* simulation. We were fortunate that this project was mapped out far in advance by the wise folks at DMG and Sugar, so we had a good amount of time to complete the R&D and creative process. As we all do in this business,

we managed to use up every last minute of the ensuing 4 months to build the best possible fluid effects and do a large amount of optimizing to get it to run as fast as possible.
The process of getting the spot done was entirely collaborative with DMG, with most of the 3D work being handled by Fusion's CG Supervisor/Co-owner, Mark Stasiuk, and DMG's creative director, Linds Redding.

"It was a very interactive process" says Linds. *"We would decide how we wanted the water to look and behave at any given point, and Mark would go away and write a piece of code that would give us the controls we needed to achieve what we wanted. We had a live Skype chat feed running constantly, and exchanged Quicktime movies and test renders via ftp daily."*

The fluid behavior evolved dramatically over time as the possibilities were explored. In the end we achieved virtually everything that the client wanted, and almost everything that was technically feasible for the current capabilities of RealFlow. In these kinds of projects, which require a lot of R&D and custom scripting work, the key is to break down the different aspects of water behavior and then step thru each aspect systematically, until you've built up the shot.

Let's walk thru the main aspects and see how they were handled:

Every Particle Has Its Place…

Any experienced RealFlow artist has tried to morph fluid to geometry – we're all familiar with the classic *"I want my logo to form out of fluid".* In RealFlow (RF) there are a couple ways to do this that are 'native', meaning they can be done with the existing package's tools and don't require custom Python scripting.

There's the Magic daemon, which is really the tool designed for this task. It works ok for very simple shapes, but there are a number of downsides to this method. When the geo gets complex, it gets very

slow. It's very difficult to get the morphed fluid to sharply define any sort of detailed geometry, tending instead to make a sort of globby approximation to the geo. Finally, it's very difficult to control the behavior as the fluid forms the shape. Typically, particles do a lot of unattractive orbiting and circulating around parts of the geo rather than settling down. All these aspects pointed us away from being able to use the Magic. The other typical way of doing a morph is to simply fill invisible geo. This is an old hack that has produced some nice results. Apart from the obvious challenges of jumping shape to shape and controlling how the transitions happen, this has the big downside of communicating to the viewer exactly what it is doing. You can see the fluid colliding with unseen barriers, and forming static, sharp edges. The result is we lose the fluid, organic look. In a lot of projects, that's a compromise we can live with or even a look that's desired, but not in the Whole project where we absolutely had to sell the idea that the water is smart and it's the thing forming the shapes, like a massive swarm of insects.

Knowing the effect was outside the scope of what RF could do natively, we set off on the path of creating a custom morph effect. Or really, a simple version of this was done as R&D just before Whole landed on our desks.

The primary challenge of getting fluid to form distinct shapes comes down to figuring out how many fluid particles you can fit on the geo, so you get a nice even coating that when meshed, will be a good copy of the shape. If the layer is overly thin, you get holes in your mesh. Put on too thick a layer and you lose details of the shape. Remember we had to define a detailed brain shape (low res = 60,000+ polys), then a detailed bridge shape including narrow vertical supports, then bananas including delicate, thin skins. So using the fluid resolution to get the physical dimensions of particles, we developed an algorithm to figure out how many particles we could fit on each poly and each poly edge of the geo. This tells you how many particles you need to capture to form the shape.

After that, it's a matter of selecting target sites on the geo for each particle, defined in terms of a kind of UV map data structure we created that's independent of the object's position, scale, rotation and

deformation (needed so the object can be animated). To give us flexibility in how the transitions happened, we built in 3 ways for particles to select targets: nearest poly with available space, farthest poly with space, or random poly with space. In the end for Whole, we went with nearest poly as that gave us the best look. This part of the code is mostly about keeping organized, as we have to track which polys and poly edges are full versus those that can take more particles. And of course, we have to keep in mind that particles could be killed off during the sim, or new ones could be added, so those have to be tracked as well.

R&D clip showing particles targeting farthest polys:

R&D clip showing particles targeting nearest polys:

The process of finding targets for particles is a slow one by nature, since you have to hunt thru a lot of geometry and do so for each particle. This was a focus of our optimization, which boiled down to exploring ways of looping, and efficient data structures, in Python. We created probably 10 different ways of doing this and measured the fastest methods. An important lesson here was: You shouldn't believe everything the Python experts say. Some methods are supposed to be fast, but aren't really at least compared to others. If you need to optimize your scripts, it's better to not make assumptions and instead, take a bit of time to run some solid tests of methods. Think about whether you can do some time consuming step just once at the beginning as a preprocessing stage, that will save you some time on every frame or substep. Only grab the global variables you need. You'd be surprised how you can save time. For the 650-frame shot in Whole, our optimizations trimmed the time per frame from about 12 minutes, to about 6 minutes. For each minute we trimmed, we could get versions done almost 11 hours faster (650 minutes = 10.8 hours). So the few days we spent optimizing won us a few weeks of time when

you consider the number of versions we had to output, with each one taking 2.7 days at 6 min/frame.

Like Bees to Honey…

The next step to getting this to work was to design forces that would pull the particles to their targets in a controllable and attractive way. A simple-minded force of attraction from the particle directly toward its target works, but is ugly. Particles get pulled on more or less straight-line paths like little robots, then overshoot their targets and start orbiting in toward their targets. A small amount of this orbiting is cool, as it gives the overall behavior a nice organic "sloshing" feel, but it's hard to control and tends to produce groups of particles that spin endlessly like little buzzing bees. To get more control, we designed a targeting force that included a pull toward target, a counter-orbiting force, and a radial variation on the force components so the fluid would get drawn horizontally until it was beneath the targeted object, and then get pulled vertically. This setup gave us control of the degree of sloshing, and also allowed us to 'sculpt' the shape of the incoming flow and make it more funnel-shaped, which was especially useful for the transition from the puddle to the brain.

R&D preview showing exaggerated particle overshoot.

We also used a force to attract particles that weren't being targeted to geometry. These non-targeting particles are ones for which there's no space on the geo. But instead of just letting them drop, the non-targeting force attracts them to the geometry center and then releases them. The end effect is a nice splashing during the transitions to the shapes. This is best seen in the transition to the bridge and that to the bananas.

Once the shape is formed, all seems well and you'd think the morphing job was done. But on seeing renders, it became immediately obvious that we wanted fluid shapes, but we had created ice sculptures. The forms were well-defined, but too static.

Getting the Water Back…

One way of regaining the fluid look is to let the particles keep sloshing around for longer as they form the shape. This turned out to make the shapes indistinct for too long, and the meshes resulting from this messy behavior looked, well, equally messy, or at least noisy. So we abandoned that idea and instead implemented two additional effects: secondary "rippling", and dripping.

First, we made the fluid ripple with a new force field. This had to be tightly controlled so we wouldn't lose the detailed shape. We tried a number of noise forces, but these just looked like noise and not very fluidal. We also tried applying deformers to the geometry, but this also looked too much like exactly that. Instead we built a force field based on noise image sequences, where the image intensity mapped to the force strength. The images were mapped to the object geometry with a spherical projection and the force direction was taken from the normal of the geometry near the particle. By altering the noise images, we could control the patterns and scale of the force, and the overall motion was still connected to the shape in an intuitive way. This gave us a fluidal motion that didn't break the definition of the shapes, and we had all the controls we needed to tweak it for the client.

R&D preview showing low-resolution fluid targeting a sphere, with strong image-based force.

R&D preview showing image-based force working on the water brain:

In isolation, the rippling force helped sell the fluid natural of the shapes, but they still needed a little something extra. Throwing ideas back and forth with DMG, Linds Redding reminded us that this project needed to sell a drink, so it was important to communicate a sense of cool, splashy freshness. Or, if you prefer, "refreshness". Imagine someone climbing out of a swimming pool on a hot summer's day, soaked with water and.... dripping. Linds mentioned that, just after each object formed, the small number of non-targeting particles that had not splashed away, and instead got caught up in the targeting fluid looked great as it dripped down.

So we set about building in extra dripping action. This turned out to be fairly simple to conceive but tricky to get to work just right. We already had the data structures in place to track particles as targeting or non-targeting, so we just needed a way to add more non-targeting particles in particular locations to get more drips. We used a "dummy" object emitter as a selection tool in this case, so we could select the faces we wanted to add drips to and then in the scripts just grabbed this poly selection -- it was a dummy emitter because it didn't emit anything. We didn't have it emit because object emitters don't add little random, tiny amounts of fluid continuously here and there like we wanted. So we just added particles on the selected polys via a script, and made these non-targeting particles so they would drip. We had to be careful how we added the particles, since exactly where we wanted them was already where our targeting particles should be. Once we could do that in a stable way, we could control the rate of emission of drips so it was a matter of just animating that in a sensible way so the drips didn't pour down like a waterfall, but did give the scene a nice refreshing feel.

R&D preview showing dripping water after the transition to the brain/

Morph to Morph to Unmorph…

The last effect we worked on was moving the fluid from one morph to another thru the shot, along with the partial release of fluid from the targeted objects. The release was needed for the last part of the shot, when two of the three bananas splash to the floor, then the last banana peels itself to reveal the banana flesh inside, then the exposed flesh releases and splashes to the floor.

Getting fluid to move from one morph to another was mostly a matter of just adding more of the same behavior, although we had to create data structures to keep track of which object was attracting particles at the moment. Mainly we had to make sure that there were enough particles on the first object to fill up the next object, and so on, since we wanted to avoid grabbing more particles from the puddle on the floor.

R&D preview demonstrating morphing between 3 objects, with a sweep-style transition

We got a simple form of this working very quickly. Once again using a "dummy" object emitter allowed us to select exactly the polys to release from. On the appointed frame, the targeting particles associated with the selected polys were converted to non-targeting particles, so they'd just fall to the floor. This worked, but it looked distinctly unattractive. One second the fluid is an active participant, in the shape of a banana while rippling and dripping, the next second that shape is falling lifelessly.

So we added a second level of complexity, making the release a little more like a melt, or really more like the shape just turns entirely into dripping fluid. On the appointed frame, we determine the subpopulation of particles that have to be released, but don't release them yet. Instead, we start releasing random clumps of them every frame so the release is gradual and the fluid drips down. This gave us a much livelier look that felt consistent with the behavior in the rest of the spot.

Work-in-progress preview in wireframe of the main shot of the commercial, showing the melt-like release of particles from the bananas.

Collaboration and Communication…

Although we rarely get the luxury of time to do this kind of R&D during projects, as important as time was the level of collaboration and communication between everyone involved. Between us and DMG, there was near continuous communication and exchange of ideas, plus the very important but not usually done admittance of where there were technical problems and limitations. We all like to tell clients *"anything you can imagine, we can do"* and while we do achieve great things in fluids, we know this is quite optimistic. On this project, we were all eager to speak freely, and that opened doors both creatively and technically to resolve issues in the best way possible. This communication extended right through to the agency Sugar and their client Fonterra. In the end, this allowed us to both push the technology to a new place, and also create the best possible product for the time and money.